Contest and Profile

**Contest System:**

**1. contests Table (Manages Contest Information)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| contest\_id | int(11) PRIMARY KEY | Unique identifier for each contest |
| title | varchar(255) | Name of the contest |
| genre | enum('Fantasy', 'Thriller Mystery', 'Youth Fiction', 'Crime', 'Horror') | Genre of the contest |
| created\_at | timestamp | Timestamp when the contest was created |

**How it works**:

* Admin or system creates contests with a title and genre.
* Users can view contests and participate in them.

**2. contest\_entries Table (Stores Contest Submissions)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| entry\_id | int(11) PRIMARY KEY | Unique identifier for each contest entry |
| contest\_id | int(11) FOREIGN KEY | Links to the contests table (which contest the entry is for) |
| user\_id | int(11) FOREIGN KEY | User who submitted the entry |
| submission\_date | timestamp | Timestamp when the entry was submitted |
| vote\_count | int(11) DEFAULT 0 | Number of votes the entry has received |
| entry\_title | varchar(255) | Title of the submitted content |
| content | longtext | The actual story or content of the entry |

**How it works**:

1. Users submit their entries for a contest.
2. The system stores details like the title, content, and number of votes.
3. Other users can vote on their entries.

**3. contest\_votes Table (Manages Voting System)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| vote\_id | int(11) PRIMARY KEY | Unique identifier for each vote |
| contest\_entry\_id | int(11) FOREIGN KEY | Links to the contest\_entries table (the contest entry the vote is for) |
| user\_id | int(11) FOREIGN KEY | User who cast the vote |
| vote\_value | tinyint(1) NOT NULL | Vote (1 = upvote, 0 = downvote) |
| created\_at | timestamp | Timestamp when the vote was cast |

**How it works**:

1. Users can vote for their favorite contest entry (upvote or downvote).
2. The system ensures each user can only vote once per entry.
3. The number of votes for each entry is updated in the contest\_entries table.

**User Profile Interaction System:**

**4. profile\_views Table (Tracks Profile Views)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| view\_id | int(11) PRIMARY KEY | Unique identifier for each profile view |
| viewer\_id | int(11) FOREIGN KEY (nullable) | User who viewed the profile (NULL if not logged in) |
| profile\_id | int(11) FOREIGN KEY | The profile being viewed |
| viewed\_at | timestamp | Timestamp when the profile was viewed |

**How it works**:

1. Every time a user visits another user's profile, the visit is logged.
2. The system records the viewer's ID (if logged in) and the profile being viewed.
3. The profile owner can see how many views their profile has received.

**5. user\_followers Table (Tracks User Follow Relationships)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| follow\_id | int(11) PRIMARY KEY | Unique identifier for the follow action |
| follower\_id | int(11) FOREIGN KEY | User who is following another user |
| following\_id | int(11) FOREIGN KEY | User being followed |
| created\_at | timestamp | Timestamp when the follow action occurred |

**How it works**:

1. Users can follow other users.
2. This table tracks who is following whom and the timestamp of the follow action.

**6. user\_books Table (Tracks Books Added by Users)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| user\_book\_id | int(11) PRIMARY KEY | Unique identifier for the user's book record |
| user\_id | int(11) FOREIGN KEY | The user who added the book |
| book\_id | int(11) FOREIGN KEY | The book that was added |
| status | enum('Reading', 'Completed', 'Dropped') DEFAULT 'Reading' | Current status of the book (e.g., reading, completed) |
| created\_at | timestamp | Timestamp when the book was added |

**How it works**:

1. Users can add books to their collection.
2. The system tracks the status of each book (reading, completed, or dropped).

**7. users Table (Stores User Information)**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Type** | **Description** |
| user\_id | int(11) PRIMARY KEY | Unique identifier for each user |
| username | varchar(50) | Unique username for login |
| password | varchar(255) | Hashed password for security |
| email | varchar(100) | Unique email address |
| role | enum('Admin', 'Regular User') DEFAULT 'Regular User' | User's role in the system |
| profile\_picture | varchar(255) | URL to the user's profile picture |
| books\_read\_count | int(11) DEFAULT 0 | Number of books the user has read |
| works\_created\_count | int(11) DEFAULT 0 | Number of books the user has created |
| created\_at | timestamp | Account creation timestamp |
| books\_listed\_count | int(11) DEFAULT 0 | Number of books listed by the user |
| followers\_count | int(11) DEFAULT 0 | Number of followers the user has |
| following\_count | int(11) DEFAULT 0 | Number of users the user is following |

**How it works**:

1. The system tracks users' personal information, including the number of books they have read, created, and listed.
2. It also tracks their followers and following counts.

**Workflow Summary:**

1. **Contests**:
   * Admin creates a contest via the contests table.
   * Users submit their entries to contests via the contest\_entries table.
   * Users vote on entries via the contest\_votes table.
   * The system ensures no self-voting and restricts voting to one per entry.
   * Entries with the most votes are considered for winner selection.
2. **User Profiles**:
   * Each time a user views another user's profile, a record is stored in the profile\_views table.
   * Users can follow other users, which is tracked in the user\_followers table.
   * Users can add books to their collection via the user\_books table.
   * The users table stores overall user information, including personal details, books read, and social connections.

All the table related to this part

CREATE TABLE `user\_followers` (

`follow\_id` int(11) NOT NULL,

`follower\_id` int(11) NOT NULL COMMENT 'User ID of the follower',

`following\_id` int(11) NOT NULL COMMENT 'User ID being followed',

`created\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp when follow action occurred'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `user\_books` (

`user\_book\_id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL COMMENT 'User who added the book',

`book\_id` int(11) NOT NULL COMMENT 'Book ID being added',

`status` enum('Reading','Completed','Dropped') DEFAULT 'Reading' COMMENT 'Reading status',

`created\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp when added'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `users` (

`user\_id` int(11) NOT NULL,

`username` varchar(50) NOT NULL COMMENT 'Unique username for login',

`password` varchar(255) NOT NULL COMMENT 'Hashed password for security',

`email` varchar(100) NOT NULL COMMENT 'Unique email address',

`role` enum('Admin','Regular User') NOT NULL DEFAULT 'Regular User',

`profile\_picture` varchar(255) DEFAULT NULL COMMENT 'Profile picture URL',

`books\_read\_count` int(11) DEFAULT 0 COMMENT 'Number of books read by the user',

`works\_created\_count` int(11) DEFAULT 0 COMMENT 'Number of books created by the user',

`created\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Account creation timestamp',

`books\_listed\_count` int(11) DEFAULT 0 COMMENT 'Number of books saved by user',

`followers\_count` int(11) DEFAULT 0,

`following\_count` int(11) DEFAULT 0

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `profile\_views` (

`view\_id` int(11) NOT NULL,

`viewer\_id` int(11) DEFAULT NULL COMMENT 'User viewing the profile',

`profile\_id` int(11) NOT NULL COMMENT 'Profile being viewed',

`viewed\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp when the profile was viewed'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `contest\_votes` (

`vote\_id` int(11) NOT NULL,

`contest\_entry\_id` int(11) NOT NULL COMMENT 'Contest entry ID the vote is for',

`user\_id` int(11) NOT NULL COMMENT 'User ID of the voter',

`vote\_value` tinyint(1) NOT NULL COMMENT 'Vote (e.g., TRUE for vote, FALSE for un-vote)',

`created\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp when vote was cast'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `contest\_entries` (

`entry\_id` int(11) NOT NULL,

`contest\_id` int(11) NOT NULL COMMENT 'Contest ID the entry is for',

`user\_id` int(11) NOT NULL COMMENT 'User ID of the participant',

`submission\_date` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp of submission',

`vote\_count` int(11) DEFAULT 0 COMMENT 'Number of votes received',

`entry\_title` varchar(255) NOT NULL COMMENT 'Name of the content',

`content` longtext NOT NULL COMMENT 'The actual content of the entry (the story or written piece).'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `contests` (

`contest\_id` int(11) NOT NULL,

`title` varchar(255) NOT NULL COMMENT 'Title of the contest',

`genre` enum('Fantasy','Thriller Mystery','Youth Fiction','Crime Horror') NOT NULL COMMENT 'Different sections for the contest',

`created\_at` timestamp NOT NULL DEFAULT current\_timestamp() COMMENT 'Timestamp when contest was created'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;